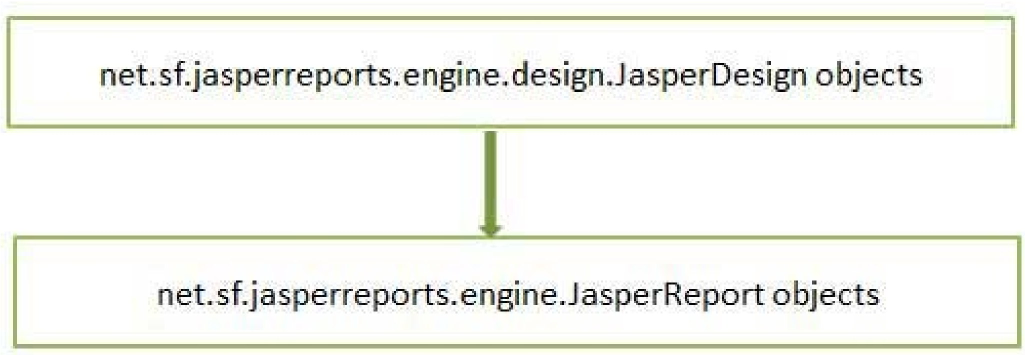
JASPER REPORT - COMPILING REPORT DESIGN

We generated the JasperReport template(JRXML file) in the previous chapter. This file cannot be used directly to generate reports. It has to be compiled to JasperReport' native binary format, called **Jasper** file. On compiling we transform JasperDesign object into JasperReport object:


net.sf.jasperreports.engine.design.JasperDesign objects
net.sf.jasperreports.engine.JasperReport objects

Interface *net.sf.jasperreports.engine.design.JRCompiler* plays a central part during compilation. This interface has several implementations depending on the language used for report expressions, which can be written in Java, Groovy, JavaScript or any other scripting language as long as compiler implementation can evaluate it at runtime. We can compile JRXML file in the following two ways:

1. Programmatic compilation.
2. Compilation through ANT task.

Programmatic compilation of JRXML

JasperReports API offers a facade class *net.sf.jasperreports.engine.JasperCompileManager* for compiling a JasperReport. This class consists of several public static methods for compiling report templates. The source of templates can be from files, input streams, in memory objects.

The contents of the JRXML file (jasper_report_template.jrxml) are as follows. It is saved at directory **C:\tools\jasperreports-5.0.1\test**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jasperReport PUBLIC "//JasperReports//DTD Report Design//EN"
"http://jasperreports.sourceforge.net/dtds/jasperreport.dtd">
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="jasper_report_template" language="groovy" pageWidth="595"
pageHeight="842" columnWidth="555" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="20">

    <queryString>
 <![CDATA[]]>
    </queryString>
    <field name="country" >
        <fieldDescription><![CDATA[country]]></fieldDescription>
    </field>
    <field name="name" >
        <fieldDescription><![CDATA[name]]></fieldDescription>
    </field>
    <columnHeader>
     <band height="23">
 <staticText>
     <reportElement mode="Opaque" x="0" y="3" width="535"
   height="15" backcolor="#70A9A9" />
     <box>
         <bottomPen lineWidth="1.0" lineColor="#CCCCCC" />
     </box>
     <textElement />
     <text><![CDATA[]]> </text>
 </staticText>
 <staticText>
     <reportElement x="414" y="3" width="121" height="15" />
     <textElement textAlignment="Center"
         verticalAlignment="Middle">
  <font isBold="true" />
     </textElement>
     <text><![CDATA[Country]]></text>
 </staticText>
 <staticText>
     <reportElement x="0" y="3" width="136" height="15" />
     <textElement textAlignment="Center"
         verticalAlignment="Middle">
  <font isBold="true" />
     </textElement>
     <text><![CDATA[Name]]></text>
 </staticText>
     </band>
    </columnHeader>
    <detail>
        <band height="16">
  <staticText>
      <reportElement mode="Opaque" x="0" y="0" width="535"
   height="14" backcolor="#E5ECF9" />
      <box>
             <bottomPen lineWidth="0.25" lineColor="#CCCCCC" />
      </box>
      <textElement />
      <text><![CDATA[]]> </text>
  </staticText>
  <textField>
      <reportElement x="414" y="0" width="121" height="15" />
      <textElement textAlignment="Center"
          verticalAlignment="Middle">
   <font size="9" />
      </textElement>
      <textFieldExpression >
          <![CDATA[$F{country}]]>
  </textFieldExpression>
  </textField>
  <textField>
      <reportElement x="0" y="0" width="136" height="15" />
      <textElement textAlignment="Center"
          verticalAlignment="Middle" />
          <textFieldExpression >
   <![CDATA[$F{name}]]>
          </textFieldExpression>
  </textField>
        </band>
    </detail>
</jasperReport>
```

The following code demonstrates compilation of the above *jasper_report_template.jrxml* file.

```
package com.tutorialspoint;

import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperCompileManager;

public class JasperReportCompile {

   public static void main(String[] args) {
      String sourceFileName = "C://tools/jasperreports-5.0.1/test" +
      "/jasper_report_template.jrxml";

      System.out.println("Compiling Report Design ...");
      try {
         /**
          * Compile the report to a file name same as
          * the JRXML file name
          */
         JasperCompileManager.compileReportToFile(sourceFileName);
      } catch (JRException e) {
         e.printStackTrace();
      }
      System.out.println("Done compiling!!! ...");
   }
}
```

Template Compilation

As next step, let's save above content in the file **C:\tools\jasperreports-5.0.1\test\src\com\tutorialspoint\JasperReportCompile.java** and import the *baseBuild.xml* in the build.xml file as below. The baseBuild.xml already has the **compile** and **run** targets:

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="JasperReportTest" default="run" basedir=".">

   <import file="baseBuild.xml"/>

</project>
```

Next, let's open command line window and go to the directory where build.xml is placed. Finally execute the command **ant -Dmain-class=com.tutorialspoint.JasperReportCompile** as follows:

```
C:\tools\jasperreports-5.0.1\test>ant -Dmain-class=com.tutorialspoint.JasperReportCompile
Buildfile: C:\tools\jasperreports-5.0.1\test\build.xml
compile:
    [javac] C:\tools\jasperreports-5.0.1\test\baseBuild.xml:27:
    warning: 'includeantruntime' was not set, defaulting to
    build.sysclasspath=last;set to false for repeatable builds
    [javac] Compiling 1 source file to C:\tools\jasperreports-5.0.1\test\classes

run:
     [echo] Runnin class : com.tutorialspoint.JasperReportCompile
     [java] Compiling Report Design ...
     [java] log4j:WARN No appenders could be found for logger
     (net.sf.jasperreports.engine.xml.JRXmlDigesterFactory).
     [java] log4j:WARN Please initialize the log4j system properly.
     [java] Done compiling!!! ...

BUILD SUCCESSFUL
Total time: 8 seconds
```

As a result of above compilation, you will see that template file *jasper_report_template.jasper* got generated in C:\tools\jasperreports-5.0.1\test directory.

Preview Compiled Report Template

The *net.sf.jasperreports.view.JasperDesignViewer*, as discussed in the previous chapter can be used to preview compiled report templates as well as JRXML templates.

To move further let's add a new target **viewDesign** to the above build.xml file, which will let us preview the compiled report. Below is the revised build.xml:

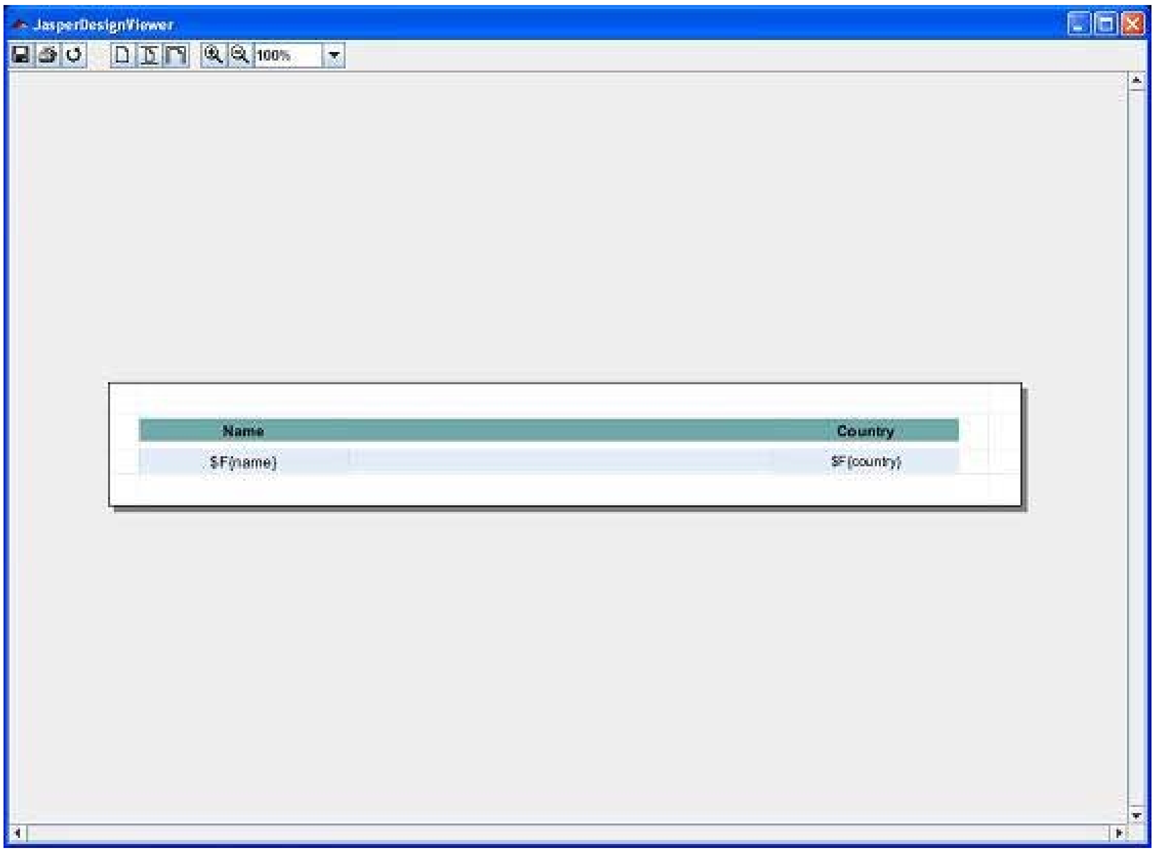
> *The import file - baseBuild.xml is picked from chapter [Environment Setup]() and should be placed in the same directory as the build.xml.*

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="JasperReportTest" default="viewDesign" basedir=".">

   <import file="baseBuild.xml" />
   <target name="viewDesign" description="Design viewer is launched
            to preview the compiled report design.">
      <java classname="net.sf.jasperreports.view.JasperDesignViewer"
            fork="true">
      <arg value="-F${file.name}.jasper" />
      <classpath ref />
      </java>
   </target>

</project>
```

Let's execute the command: **ant** (viewDesign is the default target) at command prompt. JasperDesignViewer window opens up displaying the Jasper file as below:



Compilation through ANT task

As report template compilation is more like a design time job than a runtime job, JasperReport library has a custom ANT task. For certain situations when JRXML file is created at runtime, we can't use this ANT task. The custom ANT task is called JRC and is implemented by the class: *net.sf.jasperreports.ant.JRAntCompileTask* . Its syntax and behavior are very similar to the built-in **<javac>** ANT task.

Template Compilation

Let's add new target **compilereportdesing** to our existing build.xml. Here the source folder is specified using a nested <src> tag with filesets. The nested source tag allows compiling report templates that are scattered through many different locations and are not grouped under a single root report source folder. Below is the revised build.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="JasperReportTest" default="compilereportdesing" basedir=".">
   <import file="baseBuild.xml" />
   <target name="viewDesign" description="Design viewer is launched
         to preview the compiled report design.">
      <java classname="net.sf.jasperreports.view.JasperDesignViewer"
            fork="true">
      <arg value="-F${file.name}.jasper" />
      <classpath ref />
      </java>
   </target>

   <target name="compilereportdesing" description="Compiles the JXML
         file and produces the .jasper file.">
      <taskdef name="jrc"
         classname="net.sf.jasperreports.ant.JRAntCompileTask">
         <classpath ref />
      </taskdef>
      <jrc destdir=".">
         <src>
            <fileset dir=".">
            <include name="*.jrxml" />
            </fileset>
         </src>
         <classpath ref />
      </jrc>
   </target>

</project>
```

Next, let's open command prompt and go to the directory where build.xml is placed. Execute the command **ant** (compilereportdesing is the default target) Output is follows:

```
C:\tools\jasperreports-5.0.1\test>ant
Buildfile: C:\tools\jasperreports-5.0.1\test\build.xml

compilereportdesing:
      [jrc] Compiling 1 report design files.
      [jrc] log4j:WARN No appenders could be found for logger
      (net.sf.jasperreports.engine.xml.JRXmlDigesterFactory).
      [jrc] log4j:WARN Please initialize the log4j system properly.
      [jrc] log4j:WARN See
      http://logging.apache.org/log4j/1.2/faq.html#noconfig
      for more info.
      [jrc] File :
      C:\tools\jasperreports-5.0.1\test\jasper_report_template.jrxml ... OK.

BUILD SUCCESSFUL
Total time: 5 seconds
```

File *jasper_report_template.jasper* is generated in the file system (in our case C:\tools\jasperreports-5.0.1\test directory). This file is identical to the one generated programmatically by calling the net.sf.jasperreports.engine.JasperCompileManager.compileReportToFile(). We can preview this jasper file, executing **ant viewDesign**.